UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/668,467	09/23/2003	Sachin Mullick	10830.0100.NPUS00	2942

27927 7590 05/23/2008

RICHARD AUCHTERLONIE
NOVAK DRUCE & QUIGG, LLP
1000 LOUISIANA
53RD FLOOR
HOUSTON, TX 77002

EXAMINER	
BIBBEE, JARED M	

ART UNIT	PAPER NUMBER
2161	

MAIL DATE	DELIVERY MODE
05/23/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/668,467	MULLICK ET AL.	
	Examiner	Art Unit	
	JARED M. BIBBEE	2161	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *29 January 2008*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-28,32-54 and 58-73* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-28,32-54 and 58-73* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Response to Amendment

1. This Office Action has been issued in response to amendment filed on 29 January 2008. Claims 29-31 and 55-57 have been cancelled. Claims 1-28, 32-54, and 58-73 are pending. Applicants' arguments have been carefully and respectfully considered in light of the instant amendment and are not persuasive, as they relate to the claim rejections under 35 U.S.C. 102 and 103, as will be discussed below. Accordingly, this action has been made FINAL.

Claim Rejections - 35 USC § 102

2. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

3. Claims 1-4, 10, 11, 15, 22, 32, 33-36, 44, 45, 49, and 58-73 are rejected under 35 U.S.C. 102(b) as being anticipated by Xu et al (US 6,324,581 B1).

With respect to independent claim 1, Xu teaches a method of operating a network file server for providing clients with concurrent write access to a file, the method comprising the network file server responding to a concurrent write request from a client by:

a) obtaining a lock for the file *(see column 3, line 60 through column 4, line 12)*; and then

b) preallocating a metadata block for the file *(see column 8, line 36-64)*; and then

c) releasing the lock for the file *(see column 9, lines 21-39)*; and then

d) asynchronously writing to the file *(see column 9, lines 21-39)*; and then

e) obtaining the lock for the file *(see column 3, line 60 through column 4, line 12)*; and then

f) committing the metadata block to the file *(see column 8, line 65 through column 9, line 39)*; and then

g) releasing the lock for the file *(see column 9, lines 21-39)*.

With respect to dependent claim 2, Xu further teaches a hierarchy of blocks including an inode block of metadata, data blocks of file data, and indirect blocks of metadata, and wherein the metadata block for the file is an indirect block of metadata *(see column 8, lines 36-64)*.

With respect to dependent claim 3, Xu further teaches copying data from an original indirect block of the file to the metadata block for the file, the original indirect block of the file having been shared between the file and a read-only version of the file *(see column 8, line 65 through column 9, line 39)*.

With respect to dependent claim 4, Xu further teaches concurrent writing for more than one client to the metadata block for the file *(see column 11, lines 20-37 and column 13, lines 36-42)*.

With respect to dependent claim 10, Xu further teaches writing the metadata block to a log in storage of the network file server for committing the metadata block for the file *(see column 8, line 65 through column 9, line 39)*.

With respect to dependent claim 11, Xu further teaches gathering together preallocated metadata blocks for a plurality of client write requests to the file, and committing together the preallocated metadata blocks for the plurality of client write requests to the file by obtaining the lock for the file, committing the gathered preallocated metadata blocks for the plurality of client

write requests to the file, and then releasing the lock for the file *(see column 8, line 36 through column 9, line 39)*.

With respect to independent claim 15, Xu teaches a method of operating a network file server for providing clients with concurrent write access to a file, the method comprising the network file server responding to a concurrent write request from a client by:

a) preallocating a metadata block for the file *(see column 8, line 36-64)*; and then

b) asynchronously writing to the file *(see column 9, lines 21-39)*; and then

c) committing the metadata block to the file *(see column 8, line 65 through column 9, line 39)*;

wherein the method includes gathering together preallocated metadata blocks for a plurality of client write requests to the file, and committing together the preallocated metadata blocks for the plurality of client write requests to the file by obtaining a lock for the file, committing the gathered preallocated metadata blocks for the plurality of client write requests to the file, and then releasing the lock for the file *(see column 8, line 36 through column 9, line 39)*.

With respect to dependent claim 22, Xu further teaches processing multiple concurrent read and write requests by pipelining the requests through a first processor and a second processor, the first processor performing metadata management for the multiple concurrent read and write requests, and the second processor performing asynchronous reads and writes for the multiple concurrent read and write requests *(see column 8, line 36 through column 9, line 39)*.

With respect to independent claim 32, Xu teaches A method of operating a network file server for providing clients with concurrent write access to a file, the method comprising the

network file server responding to a concurrent write request from a client by executing a write thread, execution of the write thread including:

a) obtaining an allocation mutex for the file *(see column 3, line 60 through column 4, line 12)*; and then

b) preallocating new metadata blocks that need to be allocated for writing to the file *(see column 8, line 36-64)*; and then

c) releasing the allocation mutex for the file *(see column 9, lines 21-39)*; and then

d) issuing asynchronous write requests for writing to the file *(see column 9, lines 21-39)*;

e) waiting for callbacks indicating completion of the asynchronous write requests *(see column 9, lines 21-39)*; and then

f) obtaining the allocation mutex for the file *(see column 3, line 60 through column 4, line 12)*; and then

g) committing the preallocated metadata blocks *(see column 8, line 65 through column 9, line 39)*; and then

h) releasing the allocation mutex for the file *(see column 9, lines 21-39)*.

With respect to claims 33-36, claims 33-36 correspond to claims 1-4 and are rejected for the same reasons as set forth in the rejection of claims 1-4 above.

With respect to claims 44-45, claims 44-45 correspond to claims 10-11 and are rejected for the same reasons as set forth in the rejection of claims 10-11 above.

With respect to claim 49, claim 49 corresponds to claim 15 and is rejected for the same reasons as set forth in the rejection of claim 15 above.

With respect to claim 58, claim 58 corresponds to claim 32 and is rejected for the same reasons as set forth in the rejection of claim 32 above.

With respect to dependent claim 59, Xu further teaches an uncached write interface, a file system cache and a cached read-write interface, and wherein the uncached write interface bypasses the file system cache for sector-aligned write operations *(see column 8, line 36 through column 9, line 39)*.

With respect to dependent claim 60, Xu further teaches the network file server is further programmed to invalidate cache blocks in the file system cache including sectors being written to by the uncached write interface *(see column 8, line 36 through column 9, line 39)*.

With respect to independent claim 61, Xu teaches a method of operating a network file server for providing clients with concurrent write access to a file, the method comprising the network file server responding to a concurrent write request from a client by:

a) preallocating a block for the file *(see column 8, line 36-64)*; and then

b) asynchronously writing to the file *(see column 9, lines 21-39)*; and then

c) committing the preallocated block *(see column 8, line 65 through column 9, line 39)*;

wherein the network file server also includes an uncached write interface, a file system cache, and a cached read-write interface, wherein the uncached write interface bypasses the file system cache for sector-aligned write operations, and the network file server is programmed to invalidate cache blocks in the file system cache including sectors being written to by the uncached write interface *(see column 8, line 36 through column 9, line 39)*.

With respect to dependent claim 62, Xu further teaches a final step of returning to said client an acknowledgement of the writing to the file *(see column 8, line 36 through column 9, line 39)*.

With respect to claims 63-67, claims 63-67 correspond to claim 62 and are rejected for the same reasons as set forth in the rejection of claim 62 above.

With respect to dependent claim 68, Xu further teaches a final step of saving the file in disk storage of the network file server *(see column 8, line 36 through column 9, line 39)*.

With respect to claims 69-73, claims 69-73 correspond to claim 68 and are rejected for the same reasons as set forth in the rejection of claim 68 above.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 5-9, 12-14, 16-21, 23-28, 37-43, 46-48, and 50-54 are rejected under 35 U.S.C. 103(a) as being unpatentable over Xu in view of Marcotte (US 6,449, 614 B1).

With respect to dependent claim 5, note the discussion of claim 1 above, Xu discloses all of the elements of claim 1, but fails to explicitly recite the asynchronous writing to the file includes a partial write to a new block that has been copied at least in part from an original block

of the file, and wherein the method further includes checking a partial block conflict queue for a conflict with a concurrent write to the new block, and upon failing to find an indication of a conflict with a concurrent write to the new block, preallocating the new block, copying at least a portion of the original block of the file to the new block, and performing the partial write to the new block.

However, Marcotte teaches the asynchronous writing to the file includes a partial write to a new block that has been copied at least in part from an original block of the file, and wherein the method further includes checking a partial block conflict queue for a conflict with a concurrent write to the new block, and upon failing to find an indication of a conflict with a concurrent write to the new block, preallocating the new block, copying at least a portion of the original block of the file to the new block, and performing the partial write to the new block *(see column 13, line 35 through column 14, line 43)*.

At the time of the invention, it would have been obvious to one of ordinary skill having the teachings of Xu and Marcotte before him or her, to modify the asynchronous writing of Xu to include the holding queue of Marcotte for the purpose of achieving better performance eliminate the need to suspend the system.

The suggestion/motivation for doing so would have been to manage locks and improve the system *(see column 20, lines 35-49)*.

Therefore, it would have been obvious to combine Marcotte with Xu to obtain the invention as specified in the instant claims.

With respect to dependent claim 6, note the discussion of claim 1 above, Xu discloses all of the elements of claim 1, but fails to explicitly recite the asynchronous writing to the file

includes a partial write to a new block that has been copied at least in part from an original block of the file, and wherein the method further includes checking a partial block conflict queue for a conflict with a concurrent write to the new block, and upon finding an indication of a conflict with a concurrent write to the new block, waiting until resolution of the conflict with the concurrent write to the new block, and then performing the partial write to the new block.

However, Marcotte teaches the asynchronous writing to the file includes a partial write to a new block that has been copied at least in part from an original block of the file, and wherein the method further includes checking a partial block conflict queue for a conflict with a concurrent write to the new block, and upon finding an indication of a conflict with a concurrent write to the new block, waiting until resolution of the conflict with the concurrent write to the new block, and then performing the partial write to the new block *(see column 13, line 35 through column 14, line 43)*.

At the time of the invention, it would have been obvious to one of ordinary skill having the teachings of Xu and Marcotte before him or her, to modify the asynchronous writing of Xu to include the holding queue of Marcotte for the purpose of achieving better performance eliminate the need to suspend the system.

The suggestion/motivation for doing so would have been to manage locks and improve the system *(see column 20, lines 35-49)*.

Therefore, it would have been obvious to combine Marcotte with Xu to obtain the invention as specified in the instant claims.

With respect to dependent claim 7, Marcotte further teaches placing a request for the partial write in a partial write wait queue upon finding an indication of a conflict with a

concurrent write to the new block, and performing the partial write upon servicing the partial write wait queue *(see column 13, line 35 through column 14, line 43)*.

With respect to dependent claim 8, note the discussion of claim 1 above, Xu discloses all of the elements of claim 1, but fails to explicitly recite checking an input-output list for a conflicting prior concurrent access to the file, and upon finding a conflicting prior concurrent access to the file, suspending the asynchronous writing to the file until the conflicting prior concurrent access to the file is no longer conflicting.

However, Marcotte teaches checking an input-output list for a conflicting prior concurrent access to the file, and upon finding a conflicting prior concurrent access to the file, suspending the asynchronous writing to the file until the conflicting prior concurrent access to the file is no longer conflicting *(see column 12, line 7 through column 13, line 32)*.

At the time of the invention, it would have been obvious to one of ordinary skill having the teachings of Xu and Marcotte before him or her, to modify the asynchronous writing of Xu to include the waiting list of Marcotte.

The suggestion/motivation for doing so would have been to manage locks and improve the system *(see column 20, lines 35-49)*.

Therefore, it would have been obvious to combine Marcotte with Xu to obtain the invention as specified in the instant claims.

With respect to dependent claim 9, Marcotte further teaches providing a sector-level granularity of byte range locking for concurrent write access to the file by the suspending of the asynchronous writing to the file until the conflicting prior concurrent access is no longer conflicting *(see column 12, line 7 through column 13, line 32)*.

With respect to dependent claim 12, note the discussion of claim 1 above, Xu discloses all of the elements of claim 1, Xu further teaches checking whether a previous commit is in progress after asynchronously writing to the file and before obtaining the lock for the file for committing the metadata block to the file *(see column 8, line 36 through column 9, line 39)*, but fails to explicitly recite upon finding that a previous commit is in progress, placing a request for committing the metadata block to the file on a staging queue for the file.

However, Marcotte teaches upon finding that a previous commit is in progress, placing a request for committing the metadata block to the file on a staging queue for the file *(see column 12, line 7 through column 13, line 32)*.

At the time of the invention, it would have been obvious to one of ordinary skill having the teachings of Xu and Marcotte before him or her, to modify the asynchronous writing of Xu to include the waiting list of Marcotte.

The suggestion/motivation for doing so would have been to manage locks and improve the system *(see column 20, lines 35-49)*.

Therefore, it would have been obvious to combine Marcotte with Xu to obtain the invention as specified in the instant claims.

With respect to independent claim 13, Xu teaches a method of operating a network file server for providing clients with concurrent write access to a file, the method comprising the network file server responding to a concurrent write request from a client by:

a) preallocating a block for the file *(see column 8, line 36-64)*; and then

b) asynchronously writing to the file *(see column 9, lines 21-39)*; and then

c) committing the block to the file *(see column 8, line 65 through column 9, line 39)*;

Xu fails to explicitly recite the asynchronous writing to the file includes a partial write to a new block that has been copied at least in part from an original block of the file, and wherein the method further includes checking a partial block conflict queue for a conflict with a concurrent write to the new block, and upon finding an indication of a conflict with a concurrent write to the new block, waiting until resolution of the conflict with the concurrent write to the new block, and then performing the partial write to the new block.

However, Marcotte teaches the asynchronous writing to the file includes a partial write to a new block that has been copied at least in part from an original block of the file, and wherein the method further includes checking a partial block conflict queue for a conflict with a concurrent write to the new block, and upon finding an indication of a conflict with a concurrent write to the new block, waiting until resolution of the conflict with the concurrent write to the new block, and then performing the partial write to the new block *(see column 13, line 35 through column 14, line 43)*.

At the time of the invention, it would have been obvious to one of ordinary skill having the teachings of Xu and Marcotte before him or her, to modify the asynchronous writing of Xu to include the holding queue of Marcotte for the purpose of achieving better performance eliminate the need to suspend the system.

The suggestion/motivation for doing so would have been to manage locks and improve the system *(see column 20, lines 35-49)*.

Therefore, it would have been obvious to combine Marcotte with Xu to obtain the invention as specified in the instant claims.

With respect to dependent claim 14, Marcotte further teaches placing a request for the partial write in a partial write wait queue upon finding an indication of a conflict with a concurrent write to the new block, and performing the partial write upon servicing the partial write wait queue *(see column 13, line 35 through column 14, line 43)*.

With respect to dependent claim 16, note the discussion of claim 15 above, Xu discloses all of the elements of claim 15, Xu further teaches checking whether a previous commit is in progress after asynchronously writing to the file and before obtaining the lock for the file for committing the metadata block to the file *(see column 8, line 36 through column 9, line 39)*, but fails to explicitly recite upon finding that a previous commit is in progress, placing a request for committing the metadata block to the file on a staging queue for the file.

However, Marcotte teaches upon finding that a previous commit is in progress, placing a request for committing the metadata block to the file on a staging queue for the file *(see column 12, line 7 through column 13, line 32)*.

At the time of the invention, it would have been obvious to one of ordinary skill having the teachings of Xu and Marcotte before him or her, to modify the asynchronous writing of Xu to include the waiting list of Marcotte.

The suggestion/motivation for doing so would have been to manage locks and improve the system *(see column 20, lines 35-49)*.

With respect to dependent claim 17, note the discussion of claim 15 above, Xu discloses all of the elements of claim 15, but fails to explicitly recite the network file server responding to concurrent write requests by writing new data for specified blocks of the file to the disk storage

without writing the new data for the specified blocks of the file to the file system cache, and invalidating the specified blocks of the file in the file system cache.

However, Marcotte teaches the network file server responding to concurrent write requests by writing new data for specified blocks of the file to the disk storage without writing the new data for the specified blocks of the file to the file system cache, and invalidating the specified blocks of the file in the file system cache *(see column 12, line 7 through column 13, line 32)*.

At the time of the invention, it would have been obvious to one of ordinary skill having the teachings of Xu and Marcotte before him or her, to modify the asynchronous writing of Xu to include the waiting list of Marcotte.

The suggestion/motivation for doing so would have been to manage locks and improve the system *(see column 20, lines 35-49)*.

With respect to dependent claim 18, Marcotte further teaches the network file server responding to read requests for file blocks not found in the file system cache by reading the file blocks from the file system in disk storage and then checking whether the file blocks have become stale due to concurrent writes to the file blocks, and writing to the file system cache a file block that has not become stale, and not writing to the file system cache a file block that has become stale *(see column 12, line 7 through column 13, line 32)*.

With respect to dependent claim 19, Marcotte further teaches the network file server checking a read-in-progress flag for a file block upon finding that the file block is not in the file system cache, and upon finding that the read-in-progress flag indicates that a prior read of the

file block is in progress from the file system in the disk storage, waiting for completion of the prior read of the file block from the file system in the disk storage, and then again checking whether the file block is in the file system cache *(see column 12, line 7 through column 13, line 32)*.

With respect to dependent claim 20, Marcotte further teaches the network file server setting a read-in-progress flag for a file block upon finding that the file block is not in the file system cache and then beginning to read the file block from the file system in disk storage, clearing the read-in-progress flag upon writing to the file block on disk, and inspecting the read-in-progress flag to determine whether the file block has become stale due a concurrent write to the file block *(see column 12, line 7 through column 13, line 32)*.

With respect to dependent claim 21, Marcotte further teaches the network file server maintaining a generation count for each read of a file block from the file system in the disk storage in response to a read request for a file block that is not in the file system cache, and checking whether a file block having been read from the file system in the disk storage has become stale by checking whether the generation count for the file block having been read from the file system is the same as the generation count for the last read request for the same file block *(see column 12, line 7 through column 13, line 32)*.

With respect to dependent claim 23, note the discussion of claim 15 above, Xu discloses all of the elements of claim 15, but fails to explicitly recite serializing the reads by delaying access for each read to a block that is being written to by a prior, in-progress write until completion of the write to the block that is being written to by the prior, in-progress write.

However, Marcotte teaches serializing the reads by delaying access for each read to a block that is being written to by a prior, in-progress write until completion of the write to the block that is being written to by the prior, in-progress write *(see column 12, line 7 through column 13, line 32)*.

At the time of the invention, it would have been obvious to one of ordinary skill having the teachings of Xu and Marcotte before him or her, to modify the asynchronous writing of Xu to include the waiting list of Marcotte.

The suggestion/motivation for doing so would have been to manage locks and improve the system *(see column 20, lines 35-49)*.

With respect to dependent claim 24, note the discussion of claim 15 above, Xu discloses all of the elements of claim 15, but fails to explicitly recite serializing the writes by delaying access for each write to a block that is being accessed by a prior, in-progress read or write until completion of the read or write to the block that is being accessed by the prior, in-progress read or write.

However, Marcotte teaches serializing the reads by delaying access for each read to a block that is being written to by a prior, in-progress write until completion of the write to the block that is being written to by the prior, in-progress write *(see column 12, line 7 through column 13, line 32)*.

At the time of the invention, it would have been obvious to one of ordinary skill having the teachings of Xu and Marcotte before him or her, to modify the asynchronous writing of Xu to include the waiting list of Marcotte.

The suggestion/motivation for doing so would have been to manage locks and improve the system *(see column 20, lines 35-49)*.

With respect to independent claim 25, Xu teaches a method of operating a network file server for providing clients with concurrent write access to a file, the method comprising the network file server responding to a concurrent write request from a client by:

a) preallocating a metadata block for the file *(see column 8, line 36-64)*; and then

b) asynchronously writing to the file *(see column 9, lines 21-39)*; and then

c) committing the metadata block to the file *(see column 8, line 65 through column 9, line 39)*;

Xu fails to explicitly recite the network file server responding to concurrent write requests by writing new data for specified blocks of the file to the disk storage without writing the new data for the specified blocks of the file to the file system cache, and invalidating the specified blocks of the file in the file system cache.

Xu also fails to explicitly recite the network file server responding to read requests for file blocks not found in the file system cache by reading the file blocks from the file system in disk storage and then checking whether the file blocks have become stale due to concurrent writes to the file blocks, and writing to the file system cache a file block that has not become stale, and not writing to the file system cache a file block that has become stale.

However, Marcotte teaches the network file server responding to concurrent write requests by writing new data for specified blocks of the file to the disk storage without writing the new data for the specified blocks of the file to the file system cache, and invalidating the

specified blocks of the file in the file system cache *(see column 12, line 7 through column 13, line 32)*.

Marcotte also teaches the network file server responding to read requests for file blocks not found in the file system cache by reading the file blocks from the file system in disk storage and then checking whether the file blocks have become stale due to concurrent writes to the file blocks, and writing to the file system cache a file block that has not become stale, and not writing to the file system cache a file block that has become stale *(see column 12, line 7 through column 13, line 32)*.

At the time of the invention, it would have been obvious to one of ordinary skill having the teachings of Xu and Marcotte before him or her, to modify the asynchronous writing of Xu to include the waiting list of Marcotte.

The suggestion/motivation for doing so would have been to manage locks and improve the system *(see column 20, lines 35-49)*.

With respect to dependent claim 26, Marcotte further teaches the network file server checking a read-in-progress flag for a file block upon finding that the file block is not in the file system cache, and upon finding that the read-in-progress flag indicates that a prior read of the file block is in progress from the file system in the disk storage, waiting for completion of the prior read of the file block from the file system in the disk storage, and then again checking whether the file block is in the file system cache *(see column 12, line 7 through column 13, line 32)*.

With respect to dependent claim 27, Marcotte further teaches the network file server setting a read-in-progress flag for a file block upon finding that the file block is not in the file system cache and then beginning to read the file block from the file system in disk storage, clearing the read-in-progress flag upon writing to the file block on disk, and inspecting the read-in-progress flag to determine whether the file block has become stale due a concurrent write to the file block *(see column 12, line 7 through column 13, line 32)*.

With respect to dependent claim 28, Marcotte further teaches the network file server maintaining a generation count for each read of a file block from the file system in the disk storage in response to a read request for a file block that is not in the file system cache, and checking whether a file block having been read from the file system in the disk storage has become stale by checking whether the generation count for the file block having been read from the file system is the same as the generation count for the last read request for the same file block *(see column 12, line 7 through column 13, line 32)*.

With respect to claims 37-43, claims 37-43 correspond to claims 5-9 and are rejected for the same reasons as set forth in the rejection of claims 5-9 above.

With respect to claims 46-48, claims 46-48 correspond to claims 12-14 and are rejected for the same reasons as set forth in the rejection of claims 12-14 above.

With respect to claim 50, claim 50 corresponds to claim 16 and is rejected for the same reasons as set forth in the rejection of claim 16 above.

With respect to claims 51-54, claims 51-54 correspond to claims 25-28 and are rejected for the same reasons as set forth in the rejection of claims 25-28 above.

Response to Arguments

6. Applicants' arguments with respect to objections and rejections not repeated herein are moot, as the respective objections and rejections have been withdrawn in light of the instant amendments. Those arguments that still deemed relevant are now addressed below.

A. Applicant Argues:

It is respectfully submitted that Xu fails to disclose the applicants' step (c) of releasing the lock for the file before the step (d) of asynchronously writing to the file for Xu's network file server responding to a write request from a client. Xu also fails to disclose the applicants' step (e) of obtaining the lock for the file after the step (d) of asynchronously writing to the file. Nor are applicants' steps (c) and (e) inherent or obvious from Xu because Xu's lock of column 3, line 60 through column 4, line 12 is a lock granted by a first data mover computer to a second data mover so that the second data mover may access data storage locations of the file. The lock on the file as recited in applicants' claim 1 is not a lock for accessing data storage location of the file because it is released in applicants' step (c) prior to applicant's step (d) of asynchronously writing to the file, and again obtained in applicants' step (e) after applicant's step (d) of asynchronously writing to the file. Instead, the lock on the file as recited in applicants' claim 1 is associated with the step (b) of preallocating a metadata block for the file and is associated with the step (f) of committing the metadata block to the file, because the lock on the file as recited in applicants' claim 1 is obtained in step (a) before step (b) and then released in step (c) after step (b), and it is again obtained in step (e) before step (f) and then released in step (g).

Response:

With respect to Applicant's argument, the argument is not correct and Examiner is not persuaded because Applicant's claim 1 as recited does not clearly define a sequence of seven specific steps performed in a specific order. Claim 1 at its broadest reasonable interpretation comprises seven limitations (a)-(g) which can be understood to be merely a limitation (a) then an additional limitation (b), then an additional limitation (c), etc and so forth. Nothing within the claim limits the limitations to execute one after another in a specific order. Examiner offers the suggestion of adding the claim language such as "*A method of operation a network file server for providing clients with concurrent write access to a file by a sequence of seven steps performed in a specific order as listed below, the method comprising the network file server responding to a*

concurrent write request from a client by:" or something similar to possibly help further prosecution in the manner that the Applicant is arguing.

However, claim 1's rejection is maintained as seen below: With respect to independent claim 1, Xu teaches a method of operating a network file server for providing clients with concurrent write access to a file, the method comprising the network file server responding to a concurrent write request from a client by:

a) obtaining a lock for the file *(see column 3, line 60 through column 4, line 12)*; and then

b) preallocating a metadata block for the file *(see column 8, line 36-64)*; and then

c) releasing the lock for the file *(see column 9, lines 21-39)*; and then

d) asynchronously writing to the file *(see column 9, lines 21-39)*; and then

e) obtaining the lock for the file *(see column 3, line 60 through column 4, line 12)*; and then

f) committing the metadata block to the file *(see column 8, line 65 through column 9, line 39)*; and then

g) releasing the lock for the file *(see column 9, lines 21-39)*.

B. Applicant Argues:

However, it is not understood how this passage or any other passage in Xu discloses "the original indirect block of the file having been shared between the file and a read-only version of the file."

Response:

With respect to Applicant's argument, the argument is not correct and Examiner is not persuaded because Xu does teach copying data from an original indirect block of the file to the metadata block for the file, the original indirect block of the file having been shared between the file and a read-only version of the file *(see column 8, line 65 through column 9, line 39; Note that Xu discusses the write process in column 9, lines 14-39. Specifically, Xu discloses that*

during a write a lock is obtained on the file system by disclosing the release of the lock in column 9, line 36. This suggests that the file system copy of the data is read –only to everyone other than the owner. Xu also discusses asynchronous writes, which creates a cache of read or write data, which is then written and committed to a file system, thus releasing the lock.).

C. Applicant Argues:

It is not seen where Xu discloses that writing to such an inode or indirect block would occur concurrently for more than one client instead of serially by the Owner having exclusive access to the metadata blocks of the file.

Response:

With respect to Applicant's argument, the argument is not correct and Examiner is not persuaded because Xu does teach concurrent writing for more than one client to the metadata block for the file *(see column 11, lines 20-37 and column 13, lines 36-42; Xu discloses the use of a bypass data path to allow metadata requests between second client and first file system. Xu suggests that while second client has the bypass data path to the first file system, the third client's metadata requests are forwarded between the first and second data movers allowing access to the first file system, thus demonstrating concurrency between the second and third clients.).*

D. Applicant Argues:

However, these passages fail to disclose gathering together preallocated metadata blocks for a plurality of client write requests to the file, and committing together the preallocated metadata blocks for the plurality of client write requests to the file by obtaining the lock for the file, committing the gathered preallocated metadata blocks for the plurality of client write requests to the file, and then releasing the lock for the file.

Response:

With respect to Applicant's argument, the argument is not correct and Examiner is not persuaded because Xu teaches gathering together preallocated metadata blocks for a plurality of

client write requests to the file, and committing together the preallocated metadata blocks for the plurality of client write requests to the file by obtaining the lock for the file, committing the gathered preallocated metadata blocks for the plurality of client write requests to the file, and then releasing the lock for the file *(see column 8, line 65 through column 9, line 39; Note that Xu discusses the write process in column 9, lines 14-39. Specifically, Xu discloses that metadata requests from multiple clients by data movers. Xu discloses that during a write a lock is obtained on the file system by disclosing the release of the lock in column 9, line 36. The cached read or write data is written to the nonvolatile storage of the file system and invalidating the cache thus confirming the commitment.)*.

E. Applicant Argues:

With respect to applicants' independent claim 15, as discussed above with reference to applicants' claim 11, it is respectfully submitted that Xu fails to disclose gathering together preallocated metadata blocks for a plurality of client write requests to the file, and committing together the preallocated metadata blocks for the plurality of client write requests to the file by obtaining the lock for the file, committing the gathered preallocated metadata blocks for the plurality of client write requests to the file, and then releasing the lock for the file.

Response:

With respect to Applicant's argument, the argument is not correct and Examiner is not persuaded because Xu teaches gathering together preallocated metadata blocks for a plurality of client write requests to the file, and committing together the preallocated metadata blocks for the plurality of client write requests to the file by obtaining the lock for the file, committing the gathered preallocated metadata blocks for the plurality of client write requests to the file, and then releasing the lock for the file *(see column 8, line 65 through column 9, line 39; Note that Xu discusses the write process in column 9, lines 14-39. Specifically, Xu discloses that metadata requests from multiple clients by data movers. Xu discloses that during a write a lock is*

obtained on the file system by disclosing the release of the lock in column 9, line 36. The cached read or write data is written to the nonvolatile storage of the file system and invalidating the cache thus confirming the commitment.).

F. Applicant Argues:

With respect to applicants' independent claim 32, as discussed above with reference to applicants' claim 1, it is respectfully submitted that Xu fails to disclose applicants' step c) of releasing the allocation mutex of the file [prior to the step d) of issuing asynchronous write requests for writing to the file], and applicants' step f) of obtaining the allocation mutex for the file [after the step d) of issuing asynchronous write requests for writing to the file]. In addition, the lock of Xu column 3, line 60 through col. 4, line 12 appears to be a read lock or a write lock, and not an allocation mutex. Moreover, as discussed above with reference to Xu col. 8, lines 47- 54 and Xu col. 33 line 65 to col. 34 line 9, a data mover Owner of a file does not need an allocation mutex because it has been configured to have exclusive ownership and access to the metadata blocks of the file.

Response:

With respect to Applicant's argument, the argument is not correct and Examiner is not persuaded because Applicant's claim 32 as recited does not clearly define a sequence of eight specific steps performed in a specific order. Claim 32 at its broadest reasonable interpretation comprises eight limitations (a)-(h) which can be understood to be merely a limitation (a) then an additional limitation (b), then an additional limitation (c), etc and so forth. Nothing within the claim limits the limitations to execute one after another in a specific order. Examiner offers the suggestion of adding the claim language such as "*A method of operation a network file server for providing clients with concurrent write access to a file by a sequence of eight steps performed in a specific order as listed below, the method comprising the network file server responding to a concurrent write request from a client by executing a write thread, execution of the white thread including:*" or something similar to possibly further prosecution in the manner that the Applicant is arguing.

However, claim 32's rejection is maintained as seen below: With respect to independent claim 32, Xu teaches a method of operating a network file server for providing clients with concurrent write access to a file, the method comprising the network file server responding to a concurrent write request from a client by executing a write thread, execution of the write thread including:

a) obtaining an allocation mutex for the file *(see column 3, line 60 through column 4, line 12)*; and then

b) preallocating new metadata blocks that need to be allocated for writing to the file *(see column 8, line 36-64)*; and then

c) releasing the allocation mutex for the file *(see column 9, lines 21-39)*; and then

d) issuing asynchronous write requests for writing to the file *(see column 9, lines 21-39)*;

e) waiting for callbacks indicating completion of the asynchronous write requests *(see column 9, lines 21-39)*; and then

f) obtaining the allocation mutex for the file *(see column 3, line 60 through column 4, line 12)*; and then

g) committing the preallocated metadata blocks *(see column 8, line 65 through column 9, line 39)*; and then

h) releasing the allocation mutex for the file *(see column 9, lines 21-39)*.

NOTE: With respect to claims 33, 35, 36, 45, 49, and 58, given the reasons to claims 1, 3, 4, 11, 15, and 32, respectively, Examiner maintains the rejections above.

G. Applicant Argues:

With respect to dependent claim 59, claim 59 has been amended to more clearly distinguish Xu by reciting that the network file server is further programmed to invalidate cache blocks in the file system cache including sectors being written to by the uncached write interface. It is not seen where Xu discloses that a network file server that is programmed to invalidate cache blocks in the file system cache including sectors being written to by an uncached write interface.

Response:

With respect to Applicant's argument, the argument is not correct and Examiner is not persuaded because Xu discloses that a network file server that is programmed to invalidate cache blocks in the file system cache including sectors being written to by an uncached write interface *(see column 9, lines 30-39)*.

H. Applicant Argues:

With respect to independent claim 61, claim 61 has been amended to more clearly distinguish Xu by reciting that the network file server is further programmed to invalidate cache blocks in the file system cache including sectors being written to by the uncached write interface. It is not seen where Xu discloses that a network file server that is programmed to invalidate cache blocks in the file system cache including sectors being written to by an uncached write interface.

Response:

With respect to Applicant's argument, the argument is not correct and Examiner is not persuaded because Xu discloses that a network file server that is programmed to invalidate cache blocks in the file system cache including sectors being written to by an uncached write interface *(see column 9, lines 30-39)*.

NOTE: With respect to applicants' dependent claims 5-9, 12-14, 16-21, 23-24, 37-43, and 46, these claims depend from the independent claims 1, 15, 33, and 49. Xu has been supported above with respect to the independent claims 1, 15, 33, and 49, and Marcotte does provide the

limitations of these independent claims that are missing from Xu. Therefore the rejections to dependent claims 5-9, 12-14, 16-21, 23-24, 37-43, and 46 are maintained.

I. Applicant Argues:

Arguments on pages 36-40 of Applicant arguments/remarks filed 1/29/2008 with respect to claims 5, 6, 12-14, 16-21, and 26-28.

Response:

Applicant's arguments fail to comply with 37 CFR 1.111(b) because they amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references.

J. Applicant Argues:

Action further recognizes that Xu fails to explicitly recite the network file server responding to read requests for file blocks not found in the file system cache by reading the file blocks from the file system in disk storage and then checking whether the file blocks have become stale due to concurrent writes to the file blocks, and writing to the file system cache a file block that has not become state, and not writing to the file system cache a file block that has become stale. (See, e.g., applicants' FIG. 10, steps 92, 97, 513, 98; applicants' spec., page 23 lines 5-17; page 24 lines 16-22 .) Page 19 of the Official Action cites col. 12 line 7 through column 12, line 32 for all of these limitations not explicitly recited in Xu. However, it is not understood how all of the applicants' specific claim limitations are disclosed in Marcotte column 12 line 7 through column 13, line 32.

Response:

With respect to Applicant's argument, the argument is not correct and Examiner is not persuaded because Marcotte teaches the network file server responding to concurrent write requests by writing new data for specified blocks of the file to the disk storage without writing the new data for the specified blocks of the file to the file system cache, and invalidating the specified blocks of the file in the file system cache *(see column 18, line 10 through column 20, line 17)*.

Marcotte also teaches the network file server responding to read requests for file blocks not found in the file system cache by reading the file blocks from the file system in disk storage and then checking whether the file blocks have become stale due to concurrent writes to the file blocks, and writing to the file system cache a file block that has not become stale, and not writing to the file system cache a file block that has become stale *(see column 18, line 10 through column 20, line 17)*.

NOTE: With respect to claims 37, 38, and 46-48, given the reasons to claims 5, 6, and 12-14, respectively, Examiner maintains the rejections above.

Conclusion

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to JARED M. BIBBEE whose telephone number is (571)270-1054. The examiner can normally be reached on IFP.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Apu Mofiz can be reached on 571-272-4080. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/B. S./
Examiner, Art Unit 2161

/J. M. B./
Examiner, Art Unit 2161

/Apu M Mofiz/
Supervisory Patent Examiner, Art Unit 2161